

IN-61-CR

43124

P. de

# ***Research into the Interaction Between High Performance and Cognitive Skills in an Intelligent Tutoring System***

## ***FINAL REPORT***

(NASA-CR-188821) RESEARCH INTO THE  
INTERACTION BETWEEN HIGH PERFORMANCE AND  
COGNITIVE SKILLS IN AN INTELLIGENT TUTORING  
SYSTEM Final Report (Research Inst. for  
Advanced Computer Science) 26 p

N91-32830

Unclas  
0043124

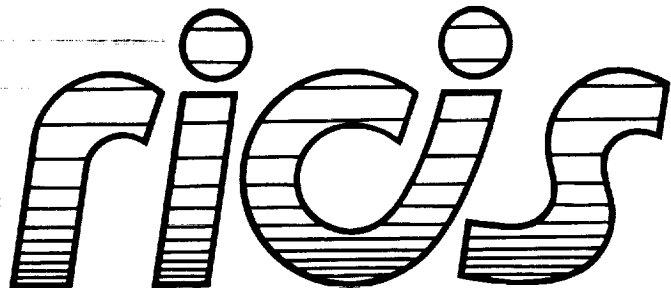
**Pamela K. Fink**

***Southwest Research Institute***

**May 31, 1991**

**Cooperative Agreement NCC 9-16  
Research Activity No. ET.23**

**NASA Johnson Space Center  
Mission Operations Directorate  
Space Station Training Office**



***Research Institute for Computing and Information Systems  
University of Houston - Clear Lake***

**T · E · C · H · N · I · C · A · L      R · E · P · O · R · T**

## ***The RICIS Concept***

The University of Houston-Clear Lake established the Research Institute for Computing and Information systems in 1986 to encourage NASA Johnson Space Center and local industry to actively support research in the computing and information sciences. As part of this endeavor, UH-Clear Lake proposed a partnership with JSC to jointly define and manage an integrated program of research in advanced data processing technology needed for JSC's main missions, including administrative, engineering and science responsibilities. JSC agreed and entered into a three-year cooperative agreement with UH-Clear Lake beginning in May, 1986, to jointly plan and execute such research through RICIS. Additionally, under Cooperative Agreement NCC 9-16, computing and educational facilities are shared by the two institutions to conduct the research.

The mission of RICIS is to conduct, coordinate and disseminate research on computing and information systems among researchers, sponsors and users from UH-Clear Lake, NASA/JSC, and other research organizations. Within UH-Clear Lake, the mission is being implemented through interdisciplinary involvement of faculty and students from each of the four schools: Business, Education, Human Sciences and Humanities, and Natural and Applied Sciences.

Other research organizations are involved via the "gateway" concept. UH-Clear Lake establishes relationships with other universities and research organizations, having common research interests, to provide additional sources of expertise to conduct needed research.

A major role of RICIS is to find the best match of sponsors, researchers and research objectives to advance knowledge in the computing and information sciences. Working jointly with NASA/JSC, RICIS advises on research needs, recommends principals for conducting the research, provides technical and administrative support to coordinate the research, and integrates technical results into the cooperative goals of UH-Clear Lake and NASA/JSC.

***Research into the Interaction Between High  
Performance and Cognitive Skills in an  
Intelligent Tutoring System***

***FINAL REPORT***

RECEIVED  
FEB 11 1964

## **Preface**

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Dr. Pamela K. Fink of the Southwest Research Institute. Dr. Glenn Freedman, Director of the Software Engineering Professional Education Center at the University of Houston-Clear Lake, served as RICIS research coordinator.

Funding has been provided by the Mission Operations Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA technical monitor for this activity was Barbara N. Pearson of the Systems/Elements Office, Space Station Training Office, Mission Operations Directorate, NASA/JSC.

The views and conclusions contained in this report are those of the author and should not be interpreted as representative of the official policies, either express or implied, of NASA or the United States Government.

INTERNATIONAL

May 31, 1991

***FINAL REPORT***

**Research into the Interaction Between  
High Performance and Cognitive Skills  
in an Intelligent Tutoring System**

**Contract No. NASA NCC9-16**

**Subcontract No. 060**

**Research Activity No. ET.23**

**SwRI Project No. 05-3515**

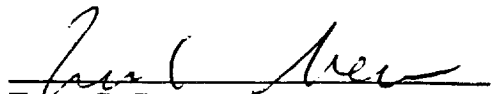
**Submitted to:**

**J. Wesley Regian, Ph.D.  
Armstrong Laboratory/IDI  
Brooks AFB, TX 78235**

**Prepared by:**

**Pamela K. Fink, Ph.D.  
Southwest Research Institute  
6220 Culebra Road  
San Antonio, TX 78228-0510**

**Approved by:**



**Terry C. Green  
Vice President  
Automation & Data Systems Division**

**INTENTIONALLY BLANK**



## TABLE OF CONTENTS

|  | PAGE |
|--|------|
| 1 BACKGROUND.....                                | 1    |
| 1.1 Intelligent Tutoring Systems.....            | 1    |
| 1.2 Types of Knowledge to be Trained.....        | 3    |
| 2 OVERVIEW OF THE CONSOLE OPERATIONS TUTORS..... | 4    |
| 2.1 The Manual Select Keyboard Tutor.....        | 5    |
| 2.2 The OMS Leak Detect Tutor.....               | 10   |
| 3 RESULTS OF THE RESEARCH.....                   | 14   |
| 4 SUMMARY AND CONCLUSIONS.....                   | 17   |
| REFERENCES.....                                  | 18   |

## LIST OF FIGURES

|  | PAGE |
|--|------|
| FIGURE 1. An example architecture for an intelligent tutoring system.....                                  | 2    |
| FIGURE 2. The curriculum model for the MSK Tutor.....  | 7    |
| FIGURE 3. The student interface to the MSK Tutor and the OMS Leak Detect Tutor.....                        | 8    |
| FIGURE 4. The curriculum tree for the OMS Leak Detect Tutor.....   | 11   |
| FIGURE 5. The leak detect procedure taught by the OMS Leak Detect Tutor.....                               | 13   |
| FIGURE 6. An architecture for the Generic Intelligent Tutoring System Implementation<br>Tool (GITSIT)..... | 15   |

## 1 BACKGROUND

Southwest Research Institute has been under contract to the Air Force Armstrong Laboratory Intelligent Systems Division (originally the Human Resources Laboratory) to develop a set of intelligent tutoring systems to study human skill acquisition. This final report discusses the research performed in the development of the two tutoring systems, summarizes the tutoring systems' capabilities, and describes the results of the work in terms of the potential for the development of a generic intelligent tutoring system shell. Though this document is to serve as a final report on just the effort involving the development of the intelligent tutoring system to train the cognitive portion of the task, the two tutoring systems are so highly interrelated that it is more valuable to include the discussion of both systems and their relationship so a picture of the entire effort can be developed.

### 1.1 Intelligent Tutoring Systems

In general, an intelligent tutoring system (ITS) can be considered to consist of four major components. A standard accepted architecture is given in Figure 1. The major components include a domain module possibly associated with a simulation, an instructional module, a student model, and a student interface. Though earlier ITS efforts tried to keep these major components separate, experience has shown that the knowledge contained in these components is highly interrelated, and that to perform each of the functions that an ITS must perform requires knowledge available in more than one component at a time. For example, teaching knowledge tends to be embodied in the student model and the intelligent interface, as well as in the instructional module, and domain expertise resides in the simulation facility and user interface, as well as in the domain module. This kind of interrelationship among sources of knowledge complicates the design and implementation of intelligent tutoring systems.

From a software engineering and artificial intelligence perspective, the design and development of such a system is similar to the design and development of four interdependent knowledge-based systems. Each of the major components needs an appropriately structured knowledge base, a corresponding inference engine, and at least one interface which may be either to a user or to one or more other knowledge-based components. Thus, for example, the domain module of an intelligent tutoring system could be considered a knowledge-based system in the domain to be taught that has interfaces to the instructional module and the intelligent student interface, while the instructional module could be considered a knowledge-based system on how to teach the subject matter that the ITS has been designed to teach that has interfaces to all of the other ITS components.

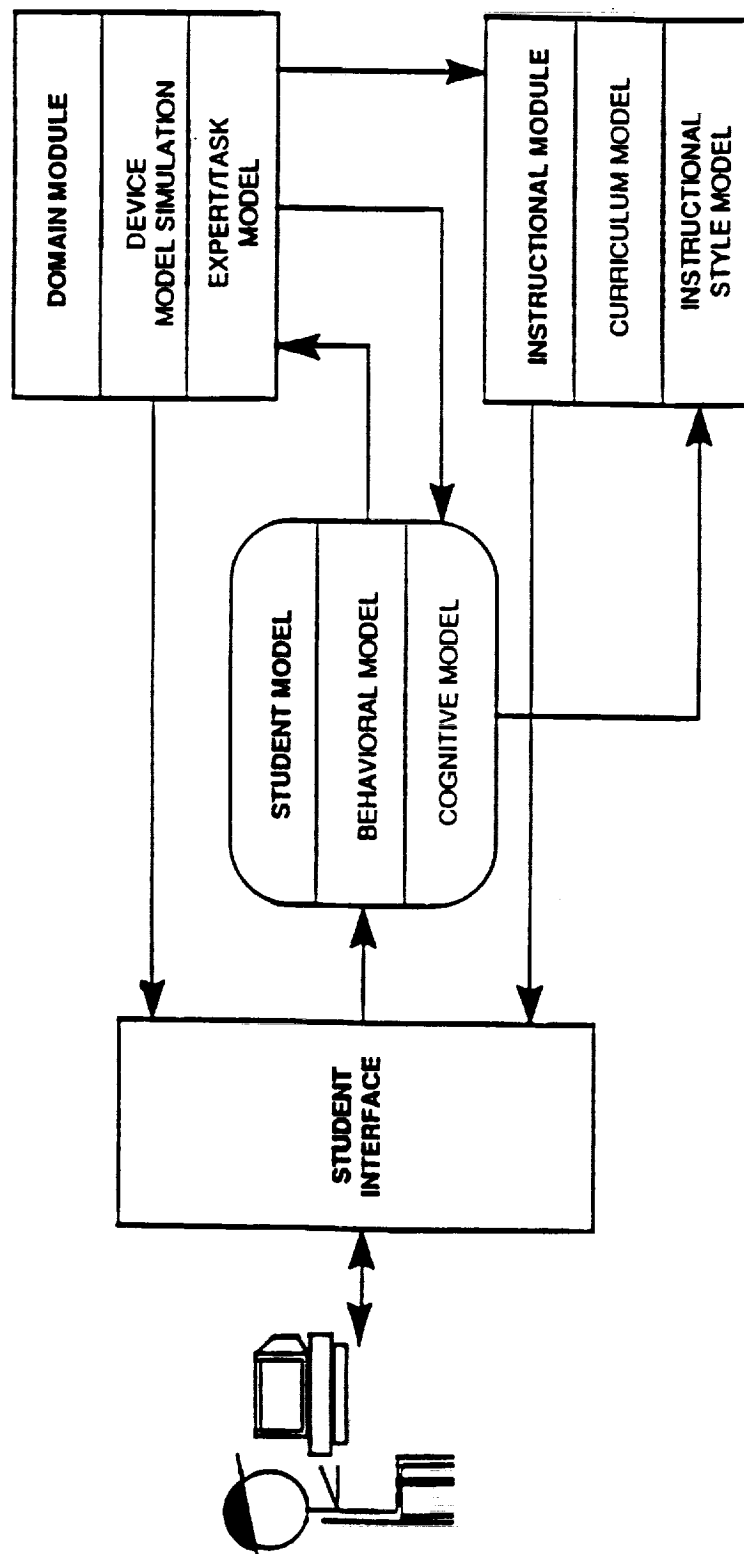


Figure 1. An example architecture for an intelligent tutoring system

## 1.2 Types Of Knowledge To Be Trained

The type of knowledge required to perform a task can range along a continuum from highly thought-oriented, or cognitive, knowledge to more physically-oriented, or skill, knowledge. The former can be termed "knowledge intensive" while the latter can be termed "high performance" (Regian and Shute, 1988). Knowledge intensive domains include such areas as medical diagnosis and practicing law. High performance skills include playing the piano, driving a car, and air traffic control. Many tasks require a combination of both types of capabilities. For example, diagnosing a patient's condition requires extensive, cognitive knowledge of medicine, but gathering the data required to make the diagnosis may require performance-oriented skill knowledge such as taking blood pressure or drawing blood.

High performance tasks are not replete of knowledge. Rather, the knowledge that is required is of a form that allows the individual to perform them "without thinking about it". This frees up cognitive processing for the performance of more knowledge intensive tasks. To carry the medical diagnosis problem one step further, a good doctor or nurse will be capable of talking to the patient or thinking about the diagnosis while taking the blood pressure or drawing the blood. At the point where cognitive processing is no longer required for performing the task, the skill is said to be "automatized" (Regian and Shute, 1988). A skill that has been automatized has the advantage of longer retention and better performance under stress --- very useful characteristics in many situations.

Most of the intelligent tutoring research to date has focused on tasks from knowledge-rich domains. For example, Anderson's Lisp Tutor (Anderson, et al. [1984]), Brown and Burton's SOPHIE system for electronic diagnosis (Brown, et al. [1982]), Carbonell and Collins' SCHOLAR system for South American geography (Carbonell [1970]), and Woolf and McDonald's MENO-TUTOR for diagnosing non-syntactic bugs in computer programs (Woolf and McDonald [1985]) all deal with domains that emphasize the conscious use of knowledge. Such tutoring systems attempt to impart certain static and/or procedural knowledge that the student is then tested on as much for the knowledge content as for the problem solving skill.

Tasks which are primarily performance-based have not attracted much intelligent tutoring research attention to date. Tutoring systems in these domains must be capable of imparting not only a certain amount of knowledge but also of drilling the student in the use of this knowledge to the point where the student need no longer concentrate on the actual problem solving task. At this point, the task is "automatized," and the individual is free to concentrate on other, more cognitively demanding issues while still performing the trained task. Testing to determine if an individual has a particular piece of knowledge is quite different from testing to determine if he/she has automatized a particular skill based on that knowledge. How such knowledge and skills should be trained also varies greatly from the approaches used in more traditional ITS domains.

Southwest Research Institute has been under contract to Armstrong Laboratory's Intelligent Systems Division to develop a pair of intelligent

tutoring systems to support research in high performance skill acquisition and its relationship to cognitive skill acquisition. In May, 1990 an intelligent tutoring system to teach a high performance skill was delivered to Armstrong Laboratory at Brooks Air Force Base. Subsequent work has built on the original work in high performance skill acquisition to include training of a more cognitively-oriented task and was delivered to Armstrong Laboratory in May 1991. The resulting intelligent tutoring system teaches a cognitive skill that is dependent on the high performance skill for successful task execution. These two tutoring systems are referred to as the Console Operations Tutors.

## 2 OVERVIEW OF THE CONSOLE OPERATIONS TUTORS

The Console Operations Tutors have been developed for Armstrong Laboratory to study the acquisition and transfer of automated skills, as well as to examine the interaction between automated and cognitive skills. The ITSs were developed to teach certain skills, both cognitively- and physically-based, that flight controllers in Johnson Space Center's Mission Control Center must be capable of performing while monitoring a Space Shuttle mission. The tutoring systems consist of one that trains a small portion of the actual operation of the Mission Control Center console and one that trains a diagnostic task that utilizes the ability to operate the console. The first tutor focuses on the use of the Manual Select Keyboard (MSK) and is called the MSK Tutor, while the latter tutor focuses on leak detection in the propulsion system of the shuttle and is called the OMS Leak Detect Tutor. The two systems are implemented in C, CLIPS, and GPR on an Apollo Domain. The CLIPS code implements most of the "intelligent" portion of the tutoring systems while the C and GPR code implements the graphics and user interface portions of the systems. The CLIPS code is fairly portable, while the C and GPR code is not.

Because the two tutoring systems were developed to provide research platforms for skill acquisition, the tutors provide additional information concerning student performance during the learning of the task. For example, the MSK Tutor maintains data on every trial performed by the student with respect to accuracy and various speeds. The OMS Leak Detect Tutor maintains accuracy and speed of each trial in the leak detect procedure as well as time durations for performance of each MSK task. In addition to the special output that can be used to support experimental research in skill acquisition, the MSK Tutor can be modified through the use of a tool that can set certain parameters within the tutor that control minimum criterion trials on various tasks and the level of difficulty of the secondary task used during the automated phase of the training. This capability provides even further flexibility for the individual interested in running skill acquisition experiments using the MSK Tutor.

Both tutors focus on the use of a modified "apprenticeship" model (Collins, Brown, and Newman, 1981) for teaching the desired skills. This model appears to be appropriate for craftsman-like, or skill-oriented tasks (Gott, 1988). This model advocates four stages of teaching/learning:

1. modelling - in which the apprentice repeatedly observes the master executing (or modelling) the target process;
2. coaching - in which the master guides and helps the apprentice during attempts to execute the process;
3. fading - where the master reduces his/her participation as the apprentice becomes able to perform the target skill;
4. reflecting - where the apprentice applies self-monitoring skills to improve his/her performance at the target skill.

This model was modified to a five-phase approach to support automated skill acquisition in the MSK Tutor as follows:

1. Static Overview Presentation, which corresponds to part of the modelling phase
2. General Procedure Overview Presentation, which corresponds to another part of the modelling phase
3. Guided Example Exercises, which corresponds to the coaching phase
4. Unguided/Speeded Example Exercises, which corresponds to the fading and reflecting phases
5. Automated Example Exercises, which is an addition to the reflecting phase for acquisition of automated skills

Each of these phases builds on the skills acquired in the previous phase. The model appears to have been fairly effective in training a skill, and training can take place to any one of the levels --- it does not have to proceed all the way to an automated level. For example, in the OMS Leak Detect Tutor, only four of the phases are used since a cognitive task is being taught that does not need to be automated. In addition, the issue of speed is not emphasized in the unguided example phase of training in the OMS Leak Detect Tutor. A brief discussion of each of the two tutoring systems is given below. Further details on the MSK Tutor can be found in Fink, 1989 and Fink and Sines, 1989.

## 2.1 The Manual Select Keyboard Tutor

The tutor that focuses on the training of the Manual Select Keyboard was delivered to HRL in June of 1990. The Manual Select Keyboard is used during initialization of the console for the ascent, orbit, and descent phases of a mission. Initialization requires the formatting of all DDD light panels, the selection of several video displays (VDT screens) to get information concerning general system status, and the selection of various voice loops to listen in on appropriate monologues and dialogues. Eventually the tutoring system could be expanded to include training on all of the various components of a Mission Control Center console, as well as a general console overview. Figure 2 provides a detail of a five-phase

training curriculum for the MSK that takes a student through static overview information, general procedure descriptions, guided example training, unguided/speeded example training, and automated example training. Each of these phases teaches a skill that builds on the skill taught in the previous phase, providing effective training for acquiring an automated skill.

The display for the tutoring system is organized into three major windows, as illustrated in Figure 3. Across the top third of the screen is a complete graphic representation of the entire console. This provides the student with an overall layout and organization of the console. The lower left half of the display provides an area where one of the panels from the console can be expanded to provide further detail. The figure shows the MSK panel. The lower right half of the screen provides the text interface where the tutor can present information, assign exercises, and accept student responses to specific verbal questions.

The graphic display of the console is mouse-sensitive. Under certain conditions it allows the student to select panels by clicking over them with the mouse to have them blown-up in the lower left window of the display. When a panel is expanded and displayed in the lower left window, it too is mouse-sensitive. A student can manipulate it by clicking the mouse over its components, thus incrementing or decrementing a thumbwheel counter, turning a push button indicator on or off, or just getting a display of the text label for the object. In this way, a large portion of the console functionality is simulated graphically in a 2-D environment and the student can gain experience in performing console operations through these simulated manipulations. The simulation provides high cognitive fidelity, but lower physical fidelity.

Training on the use of the MSK proceeds through the five phases discussed above and shown in Figure 2. The first phase of training on the MSK, the static overview presentation, provides an overview of the MSK layout and structure. The MSK panel is expanded in the lower left window on the screen and the system steps through each of its functional components, highlighting them on the graphics display and describing them with text in the lower right window. This particular phase is illustrated in Figure 3, where the mode select push button indicators are highlighted in the graphics on the left and their description appears in the text on the right. A student can move forward and back at his/her own pace through this portion of the tutorial. At the end, the student must pass an identification test where the student is asked to click over the various components of the console to indicate his/her response to the tutoring system's questions in order to proceed on to the next phase of the training. Based on the score, the student is allowed to move on or required to review the material.

Manipulation of the MSK can take place in one of five modes, selected with the push button indicators in the upper right corner of the MSK panel. The procedure for manipulating the MSK varies depending on the mode selected, so the student's training consists of five general procedures to be mastered. Because all of the objects on the MSK panel remain the same for each procedure, the first static overview phase applies to all procedures. However, at the procedural overview phase, the training tree branches to allow the student to concentrate on learning one of the



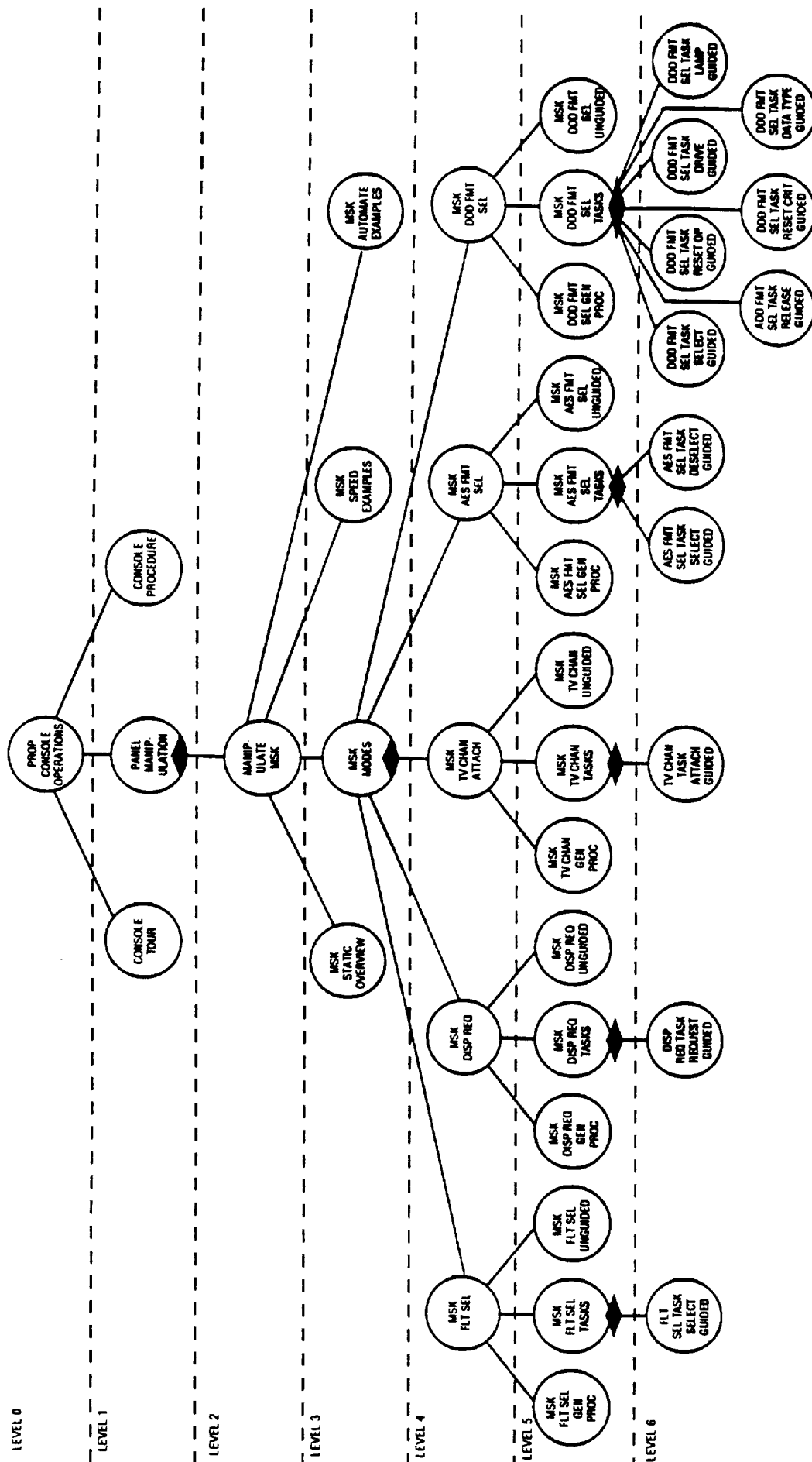


Figure 2. The curriculum model for the MSK Tutor

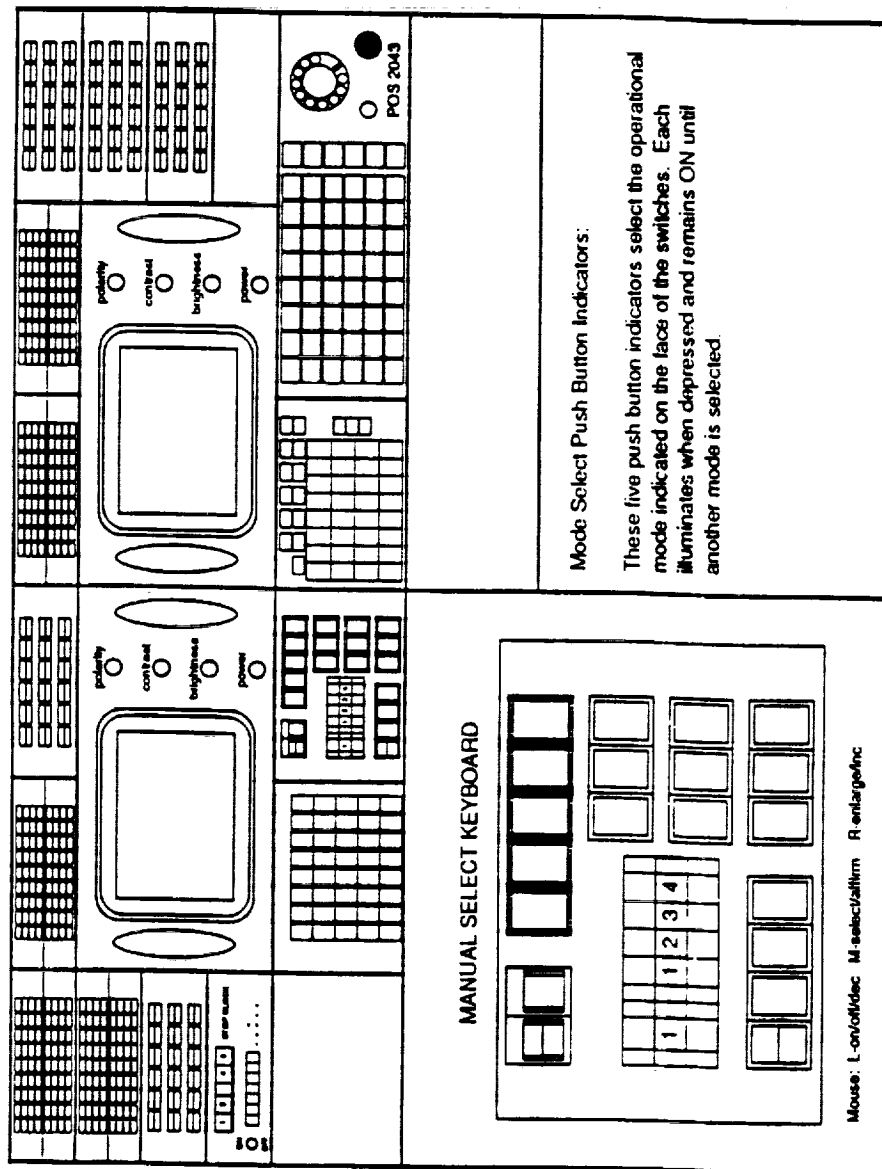


Figure 3. The student interface to the MSK Tutor and the OMS Leak Detect Tutor

procedures at a time. A task selection node called "MSK modes" is used to select the mode of operation and phases 2 and 3 of training are then subsumed under each independent task (see Figure 2). As a result, the second phase of training, namely General Procedure Presentation, provides an overview of the procedural process for manipulating the MSK in one single mode, such as display request mode. The presentation of the mode is accomplished in a manner similar to the MSK overview, using a verbal description.

Once a general description of the procedure for a mode is given, the third phase of the training, Guided Example Exercises, demonstrates specific examples of the procedure to the student by generating problems in the given mode and solving them visually on the screen. Then the student is "tested" by requiring that he/she perform the procedure on examples generated by the tutoring system. The system will prompt the student at each step and verify its correctness before moving on to the next step. If an error occurs, rules based on the error that the student has just made and the student's history of errors are used to coach the student to perform the procedure correctly. Satisfactory performance in this phase of training is based mainly on accuracy, but speed is also considered. When a student has consistently performed the assigned exercises with complete accuracy and the speed of performance has more or less plateaued, then the system allows the student to move on to the next phase of training.

The fourth phase of training, Unguided/Speeded Example Exercises, no longer guides or coaches the student through the exercises. Instead, the system simply presents an exercise, again concentrating on the same mode of MSK operation as in the Guided Example Exercises, and the student must manipulate the MSK appropriately with the mouse to achieve the requested action. Feedback to the student is limited to whether or not they performed the task correctly, and what steps in the procedure they did right and wrong. If accuracy becomes a problem, then the student is remediated back to the Guided Example Exercises.

The fourth phase of the training wraps up with a cumulative lesson where tasks from all modes are given in random order for the student to practice. The system at this point watches which modes of MSK operation the student is having trouble with, so that if remediation is necessary, it will be to the appropriate mode. Based on consistently performing with complete accuracy and reaching a point where speed is no longer improving significantly, the system then allows the student to move on to the final phase of the training.

The final phase is a repeat of the fourth phase only with an additional task that must be performed simultaneously by the student while doing the assigned exercise. While requesting a particular video display or formatting a set of DDD lights, the student must also acknowledge certain patterns of beeps by hitting the appropriate function key. The system assumes that the student has successfully automatized the MSK manipulation process when the accuracy in performing both tasks has reached one hundred percent and the speed of performing the assigned exercise and responding to the beeps has reached a peak for that particular student.

It is important to note that during the final three phases of training, where skill is being acquired and tested, no predetermined number

of trials is used to determine whether or not the student should move on. Advancement to the next phase in training depends on the particular student's performance. Though accuracy is required to be one hundred percent correct, ultimate speed can vary based on the student. The system looks for the student's leveling off in order to determine when to move on. The decision to backup and review material is based on how much difficulty the student is having attaining the required perfect accuracy. Remediation can backup selectively based on student errors and even backup all the way to the start of the training program if necessary. In this way the system can be used to refresh the memories of individuals who have been interrupted in their training for a period of time, as well as those who are seeing the material for the first time.

## 2.2 The OMS Leak Detect Tutor

The OMS Leak Detect Tutor was delivered to HRL in May of 1991 under the contract for which this document is the final report. It was designed to teach a student how to examine the data available through the propulsion console to determine the status of the helium portion of the Orbital Maneuvering System (OMS). This is basically a diagnostic task and is, therefore, cognitively-oriented. The task, however, depends on the physically-oriented skill of operating the Manual Select Keyboard (MSK) in order to obtain the necessary data. The tutoring system to train the leak detection task must, therefore, know about both skills in order to evaluate student performance and recommend appropriate teaching actions.

As a result, the tutoring system for training leak detection in the helium portion of the OMS was built based on the original MSK Tutor. When requesting data screens and examining the state of the various relevant DDD lights, the student can interface with the same simulation as was provided in the MSK Tutor. In the OMS Leak Detect Tutor, however, the actual effect of performing the action on the MKS is apparent. For example, requesting that a VDT display be brought up on one of the display screens results in that VDT display appearing on the tutor's display, thus allowing the student to actually view the data. In addition, drawings representing the shuttle system of interest, namely the helium portion of the OMS, are available and the student is taught the various components of the system and their functions and interrelationships.

The OMS Leak Detect Tutor teaches the student how to perform leak detection on the helium portion of the OMS through several major phases. A curriculum diagram, much like the one used to teach MSK operations, can be drawn to illustrate what is taught when during a tutoring sequence, as shown in Figure 4. Similar modes of training, such as static overview, guided examples, and unguided examples are used to teach the leak detect procedure as was used to teach the MSK operations. However, the static overview portion of the training is much more complex than that of the MSK Tutor. In addition, issues of speed and automated skill were not so important, so those aspects are not explicitly trained for in the OMS Leak Detect Tutor.

As can be seen in Figure 4, the first phase of training involves a



static overview of the helium system. This static overview can be broken down into several major components, all related to a type of information that the student needs to know in order to perform the ultimate task of leak detection. These major components are the Space Shuttle helium system overview, the relevant VDT displays, the relevant DDD lights, and the system functionality in various working and failure modes. The Space Shuttle helium system is presented in three levels of detail ending with a graphic depiction of how the helium tanks, valves, temperature sensors, pressure sensors, etc. are organized and related. The VDT displays are presented graphically just as they appear to the flight controller in Mission Control and the relevant data points are highlighted and discussed. The DDD lights that are relevant are highlighted one at a time and discussed as well. The final portion of the static overview takes the student through all of the VDT screens and DDD lights describing what happens to the various data points for normal operations (coast mode) and seven failure modes that can be manifested as an apparent helium system leak. During each of these discussions, an example scenario is generated that provides a means of modifying the data so that the student can become familiar with the various ways that the data, such as pressure sensor and temperature sensor readings, change based on the particular failure mode being illustrated. After each major component of information, the student must pass a multiple choice test before going on to the next area.

Once the student has achieved a basic knowledge of the Space Shuttle helium system and the relevant data available through VDT displays and DDD lights for monitoring leaks, the tutoring system then provides some practice in the use of the MSK to perform the tasks needed to bring up VDT displays, examine DDD lights, and format the console. This serves as a refresher in the skills that the student learned when going through the MSK Tutor. The student is drilled in the needed skills until proficiency is demonstrated.

Once the student has acquired all of the necessary static knowledge and the tutoring system is fairly confident of the student's skill with respect to running the MSK, the student has all of the skills needed to perform the leak detection task except an understanding of what steps to perform when. The final phase of training provides a general procedure overview, some guided example exercises broken down by the specific type of failure, and then some unguided example exercises that mix up the various failure modes. In the tutoring system, the general procedure is presented verbally, using a randomly generated scenario as the problem to be solved. Figure 5 provides the fault tree for the diagnostic process. During the guided examples phase, each type of failure mode is drilled until proficiency is achieved. Problems are presented to the student through the generation of scenarios that are manifested as appropriate modifications to the data on the VDT screens and DDD lights. The student is prompted and provided feedback at each step in the procedure. During the unguided examples phase, students are again provided exercises that are scenarios representing particular failure modes. However, the failure modes are selected randomly and no feedback is given during the performance on a problem. Feedback is held until the end and summarized. The guided examples phase works on student accuracy, while the unguided examples phase works on efficiency, including which VDT screen is best to look at for a particular piece of data, etc.

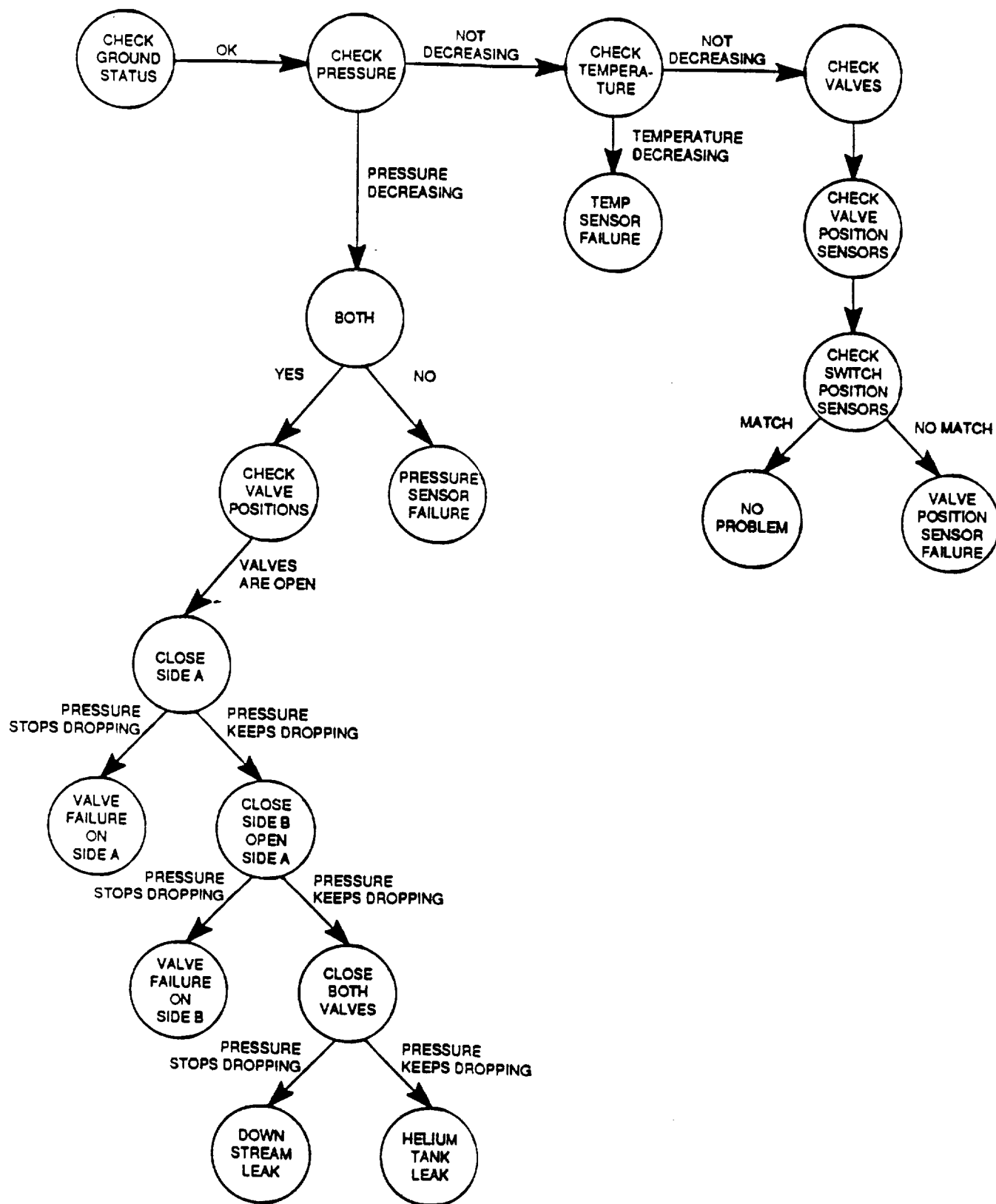


Figure 5. The leak detect procedure taught by the OMS Leak Detect Tutor

It should be noted that there is no absolute with respect to student performance on the leak detection task. The student must perform the task accurately, but speed is relative. Unlike the MSK Tutor, the OMS Leak Detect Tutor does not explicitly examine time with respect to performance (though this data is saved in the student's performance file for further analysis by the experimenter). Rather, the scenarios that are generated for each exercise are set-up to run for a maximum of four minutes. If the student cannot perform the task within the allotted time, then they are marked as incorrect and given another trial, or remediated, depending on the student's specific performance. As a result, the system provides flexible one-on-one training for a task that entails both physically- and cognitively-oriented skills.

### 3 RESULTS OF THE RESEARCH

Historically, the development of intelligent tutoring systems has been primarily based on implementation of software from scratch. That is, the code to implement an intelligent tutoring system has usually been written in a programming language that is more general than one designed specifically for implementing ITSs. As a result, development of an ITS is expensive and time consuming. Because of the similarities between the components of an ITS and the architecture of knowledge-based systems, we believe that ITS development could benefit from what has been learned in the area of knowledge-based system development and the use of knowledge-based system development tools. It is possible, with current knowledge-based system development technology, to design and implement an intelligent tutoring system development tool.

The two intelligent tutoring systems that the Institute has delivered to Armstrong Laboratory over the past year are unique in the design and implementation principles employed in their development. Though one teaches a fundamentally physical skill to an automated level and the other teaches a fundamentally cognitive skill, they both are based on the same code. The current software that implements the two tutoring systems embodies a number of data structures and software modules that constitute an approach to the implementation of an intelligent tutoring system for training skill-based and cognitive tasks. Figure 6 illustrates how the two systems are fundamentally organized.

The key attributes of this system architecture involve the use of static data structures to represent the curriculum, the expert model, and the student model. These data structures are trees and/or graphs that provide a natural way to represent hierarchical and sequential knowledge concerning processes. A process in this representation can be a sequence of skills that need to be mastered and that are associated with a means of teaching each skill, as is the case with the curriculum model, or it can be a sequence of steps that should be performed that represents one of the skills to be mastered, as is the case with the expert model. The student model is an overlay of the curriculum model annotated to indicate the student's progression through that model, plus several lists that contain data on actual performance compared with the expert model for each of the tasks to be mastered.



## THE TUTORING SHELL

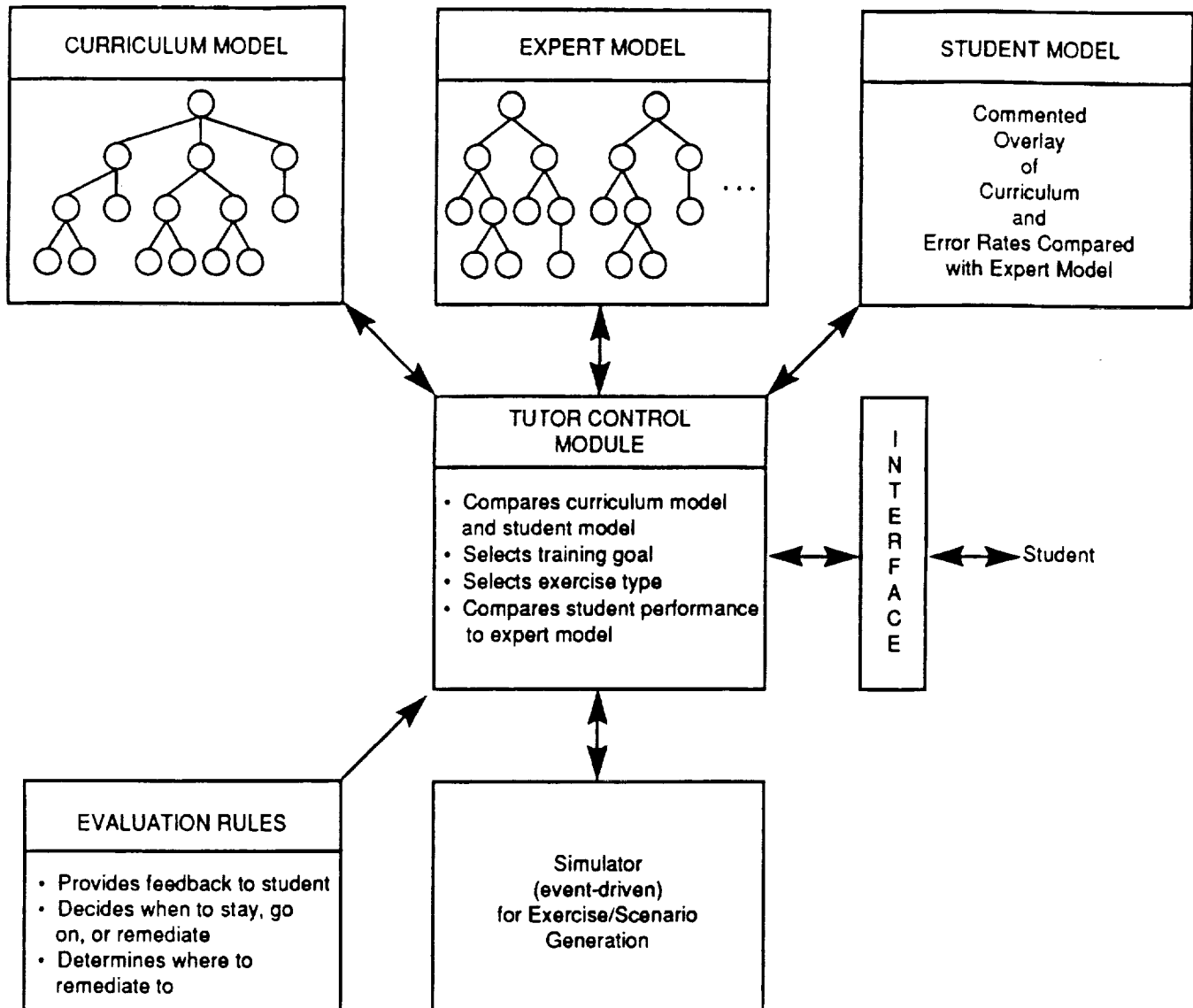


Figure 6. An architecture for the Generic Intelligent Tutoring System Implementation Tool (GITSIT)

The various modules within this tutoring system architecture interpret and act on the data maintained in the curriculum, expert, and student model data structures. Modification of the expert and curriculum data structures alters the knowledge that is taught to the student and the instructional strategy used to teach it. The student model is generated automatically based on the curriculum model and the student's performance with respect to the expert model.

The curriculum model data structure maintains the instructional strategy. This data structure can be designed to provide various instructional approaches. For example, to provide a rote learning mode, the curriculum would contain static overview nodes for all information to be presented. Based on this curriculum, the student would then be presented information through text and/or visual displays and/or audio and then tested through a multiple choice/identification type test. Alternatively, a practice-based instructional strategy would use only the guided and/or unguided example phases of training. A learning by examples mode of teaching would entail removing the static overview phase and using the system's simulation capabilities and its model of expertise to work through examples for the student. The system functionality phase of the OMS Leak Detect Tutor illustrates how the system can teach system functionality by example. Finally, the console overview portion of the MSK Tutor demonstrates how the tutoring system allows the student to learn about the propulsion console through free-play, discovery learning.

The expert model embodies the domain knowledge to be taught. Altering the expert model data structure can alter both the expert knowledge to be taught by the system as well as the type of knowledge. For example, the expert model can contain low level propositional information about the domain, such as the knowledge presented in the static overview phase in the existing tutors. Or, the expert model can teach a functional model of a device, such as the knowledge embedded in the system functionality phase of the OMS Leak Detect Tutor. Finally, it can also teach procedural knowledge, such as the sequence of steps that a student should perform to run the MSK or to troubleshoot the helium tank portion of the OMS.

The Tutor Control Module illustrated in Figure 6 has the capability of using the student model to support a search of the curriculum model to determine where in the curriculum to provide training for the given student. Once the appropriate location in the curriculum is found, a training goal is generated, such as guided examples for the DDD Format Select Mode of the MSK. This training goal can then be associated with a particular type of task and a specific expert model for the task. Exercises for the selected task to be trained can then be generated randomly. As a result, the student will most likely never see the exact same problem more than once when being trained on a skill.

Once an exercise type has been selected and a specific exercise is generated, the tutoring system presents the problem to the student and monitors the student's actions. Presentation is performed through the student interface, but monitoring is done based on the expert model. Feedback to the student is then dependent on the student's performance on the exercise and the current mode of training. For example, if the student is in guided example mode, then step-by-step prompting is provided along with feedback on incorrect performance immediately. The feedback is

designed to coach the student into the correct behavior. If the student is in unguided example mode, however, no prompting is provided and feedback is only provided after the entire task is completed. This feedback is just oriented towards what was done right and what was done wrong. No real coaching takes place. Thus, the expert model is used to determine the subject matter content of the feedback while the curriculum model is used to determine the type and timing of the feedback.

Based on student performance over a set of exercises, a set of rules referred to as remediation rules are used to determine if the student's performance indicates a need for additional exercises on the current training goal, if a previous training goal needs to be revisited (remediated), or if a new training goal should be found that will progress the student towards completion of the training curriculum. The Tutor Control Module then uses this decision to guide the next round of search through the curriculum model for the training goal that will drive the next exercise presentation to the student.

The basic architecture illustrated in Figure 6 is embodied in both of the tutoring systems for training tasks associated with the Mission Control Center console. The current implementation is not as clean as is illustrated in Figure 6, but all of the functionality is there. When the MSK Tutor software was used to develop the OMS Leak Detect Tutor, the main changes that were required to the tutoring system code were related to expanding how static overview training is handled and how remediation and feedback is handled. Of course, new curriculum and expert models were also required. We believe that the work performed on the two Mission Control Center Console tutors provides an excellent basis for generalizing to a generic ITS implementation tool for many training tasks.

#### 4 SUMMARY AND CONCLUSIONS

Southwest Research Institute has developed two intelligent tutoring systems for Armstrong Laboratories. These tutoring systems are being used to study the effectiveness of intelligent tutoring systems in training high performance tasks and the interrelationship of high performance and cognitive tasks. The two tutoring systems, referred to as the Console Operations Tutors, were built using the same basic approach to the design of an intelligent tutoring system. This design approach allowed researchers at SwRI to more rapidly implement the cognitively-based tutor, namely the OMS Leak Detect Tutor, by using the foundation of code generated in the development of the high performance-based tutor, namely the MSK Tutor. We believe that the approach can be further generalized to develop a generic intelligent tutoring system implementation tool.

## References

- Anderson, J., Farell, R., and Sauers, R. (1984). Learning to Program in LISP. In Cognitive Science, 8 (pp. 87-129).
- Bloom, B.S., 1984, "The Two-Sigma Problem: The Search for Method of Group Instruction as Effective as One-to-One Tutoring," Educational Research, June-July, pp. 3-15.
- Brown, J.S., Burton, R.R., and deKleer, J. (1982). Knowledge Engineering and Pedagogical Techniques in SOPHIE I, II, and III. In D. Sleeman and J.S. Brown (Eds.), Intelligent Tutoring Systems. London: Academic Press.
- Carbonell, J.R. (1970). AI in CAI: An Artificial Intelligence Approach to Computer-Aided Instruction. In IEEE Transactions on Man-Machine Systems. MMS-11(4) (pp. 190-202).
- Collins, A., Brown, J.S., and Newman, S.E., 1987, "Cognitive Apprenticeships: Teaching the Craft of Reading, Writing and Mathematics," in L.B. Resnick (ed.), Cognition and Instruction: Issues and Agendas, Lawrence Erlbaum: Hillsdale, N.J.
- Fink, P.K., 1989, "Issues in Representing Knowledge for Training High Performance Skills," Proceedings of the Second Intelligent Tutoring Systems Research Forum, San Antonio, TX, April.
- Fink, P.K., and Sines, L.J., 1989, "An Intelligent Tutor for a High Performance Domain," in Proceedings of the AIAA Computers in Aerospace VII Conference, Monterey, CA, October 3-5, pp. 572-580.
- Gott, S.P., 1988, "Apprenticeship Instruction for Real-World Tasks: The Coordination of Procedures, Mental Models, and Strategies," in E.V. Rothkopf (ed.), Review of Research in Education, 15, American Educational Research Association: Washington, D.C.
- Kyllonen, P.C. and Shute, V.J., 1988, "A Taxonomy of Learning Skills," in P. Ackerman, R. Sternberg, and R. Glaser (eds.), Learning and Individual Differences, Freeman: San Francisco.
- Regian, J.W., and Shute, V.J., 1988, "AI in Training: The Evolution of Intelligent Tutoring Systems," Proceedings of the Conference on Technology and Training in Education, Biloxi, MS.
- Woolf, B., and McDonald, D.D. (1985). Building a Computer Tutor: Design Issues. In AEDS Monitor, 23(9-10) (pp. 10-18).